

Mastère Spécialisé®
SMART SYSTEMS & IoT

Laboratoire ELLIADD

Tuteur de l'école :
Mme Besma ZEDDINI
Enseignante chercheur en
Intelligence Artificielle & Cybersécurité

Tuteur du laboratoire :
M. Thibaud HULIN
Enseignant chercheur, MCF HDR en
Sciences de l'Information et
de la Communication

Développement d'Ontologie et de Plate-Forme Sémantique

ZHANG Hao



Remerciements

Avant tout développement sur cette expérience professionnelle, je tiens à remercier tous les membres du projet HUMANités Numériques pour l'Éducation (HUMANE) qui m'a ont accueilli sur Discord. Merci à eux pour leur dévouement au projet HUMANE, car le corpus qu'ils ont généré est la base de mon travail dans ce stage.

Je tiens à remercier tout particulièrement mon tuteur du laboratoire de stage, Monsieur Thibaud HULIN, pas seulement pour m'avoir accepté par ce stage, aussi pour avoir pris pas mal de temps pour m'expliquer les notions principales dans le domaine des Humanités Numériques et Éducation (HNE) et discuter avec moi pour le choix de la plate-forme sémantique et le choix de base de données. Pendant le stage, il m'a donné beaucoup de liberté pour choisir une plate-forme sémantique et une base de données, et beaucoup de temps libre pour acquérir de nouvelles compétences que je ne maîtrisais pas à l'époque, notamment les compétences liées à l'ontologie, Neo4j et Flask. Comme mon stage est à distance, la plupart de mon travail doit être fait consciemment. Je suis très reconnaissant à M. HULIN pour sa confiance et son soutien.

Je remercie Monsieur Cédric LELU, pour la permission et le support technique opportun pour mon utilisation du serveur distant du laboratoire.

Aussi je tiens à remercier mon tuteur d'école, Mme Besma ZEDDINI, ainsi que toutes les personnes m'ayant permis de profiter d'un stage aussi intéressant et complet.

Résumé

Du 3 janvier 2022 au 3 juin 2022, j'ai eu la chance d'effectuer un stage au sein du Laboratoire ELLIADD à distance et de travailler avec les membres du Projet HUMANE.

Au cours de ce stage et à travers ma mission du développement d'ontologie et de plate-forme sémantique, j'ai pu découvrir et apprendre le Knowledge Graph et le web sémantique, ainsi que comment utiliser les programmes Python pour obtenir des documents sous forme de XML, comment utiliser les programmes Python pour construire des ontologies, construire des bases de données Neo4j et des pages web Flask, et comment obtenir les métadonnées du service web via OAI-PMH via le programme Python. En conclusion, grâce à ce stage, j'ai acquis une compréhension de base et une expérience pratique des concepts et des opérations de l'ontologie, du Knowledge Graph et du web sémantique, ce qui a élargi mes perspectives de carrière. Non seulement j'ai saisi l'occasion d'apprendre à extraire du contenu pertinent via XML et d'autres types de fichiers, mais j'ai également appris à exploiter et à créer des bases de données Neo4j et à créer des pages web via Flask. J'ai fourni un soutien plus professionnel pour mon développement futur.

Le contenu de ma pratique cette fois est principalement divisé en deux parties, l'une est la composition de l'ontologie, et l'autre est la construction du web sémantique lié à l'ontologie. Dans une première partie, j'utiliserai le programme Python pour obtenir les informations dans les métadonnées du service web en extrayant la méthode OAI-PMH ou en extrayant les informations dans le fichier XML converti à partir de la carte mentale pour construire l'ontologie. Et une fois l'ontologie construite, elle est encore modifiée et enrichie. Dans la deuxième partie, la même méthode est utilisée pour obtenir des informations, et ces informations sont converties en base de données Neo4j selon le même format de l'ontologie précédente, et une plate-forme sémantique de base est construite via le framework Flask.

Mots-clés : Python, web Sémantique, Ontologie, Neo4j, Flask

Summary

From January 3, 2022, to June 3, 2022, I had the chance to do an internship at the ELLIADD Laboratory remotely and to work with the members of the HUMANE Project.

During this internship and through my mission of ontology and semantic platform development, I was able to discover and learn the knowledge graph and the semantic web, as well as how to use Python programs to obtain documents in the form of XML, how to use Python programs to build ontologies, build Neo4j databases and Flask web pages, and how to get web service metadata via OAI-PMH through Python program. In conclusion, through this internship, I gained a basic understanding and experience of the concepts and operations of ontology, knowledge graph and semantic web, which broadened my career prospects. Not only did I take the opportunity to learn how to extract relevant content via XML and other file types, but I also learned how to exploit and create Neo4j databases and create web pages via Flask. I provided more professional support for my future development.

The content of my practice this time is mainly divided into two parts, one is the composition of the ontology, and the other is the construction of the semantic web related to the ontology. In a first part, I will use the Python program to get the information in the metadata of the web service by extracting the OAI-PMH method or by extracting the information in the XML file converted from the mind map to build the ontology. And once the ontology is built, it is further modified and enriched. In the second part, the same method is used to obtain information, and this information is converted into a Neo4j database according to the same format as the previous ontology, and a basic semantic platform is built through the Flask framework.

Keywords : Python, Semantic web, Ontology, Neo4j, Flask

Table des matières

1 Introduction	6
2 Monographie de l'entreprise	7
2.1 Présentation du laboratoires	7
2.2 Recherches du laboratoire ELLIADD	7
2.2.1 Pôle « Arts et Littérature » (AL)	7
2.2.2 Pôle « Conception, Création, Médiations » (CCM)	7
2.2.3 Pôle « Contextes, Langages, Didactiques » (CLD)	8
2.2.4 Pôle « Discours, Texte, Espace Public, Société » (DTEPS)	8
2.2.5 Pôle « ERgonomie et COnception des Systèmes » (ERCOS)	8
2.3 Contexte de la recherche connexe	9
2.3.1 Formation du projet HUMANE	9
2.3.2 Le but du projet HUMANE	9
2.3.3 Coopération du projet HUMANE	9
2.3.4 Intégration du projet HUMANE au sein du laboratoire ELLIADD	10
2.4 Contexte et objectifs du stage	10
2.4.1 Contexte du stage	10
2.4.2 Objectifs du stage	10
3 Mes missions	12
3.1 Informations générales sur ce stage	12
3.1.1 Tâches à effectuer	12
3.1.2 Modèle de stage	12
3.2 Choix de la plate-forme sémantique	13
3.2.1 Whyis ^[9] , un cadre de graphe de connaissances à l'échelle nanométrique	13
3.2.1.1 Graphique des connaissances à l'échelle nanométrique	14
3.2.1.2 Architecture de Whyis	15
3.2.1.3 Mes considérations	16
3.2.2 Semantic MediaWiki ^[15]	16
3.2.2.1 SMW utilise le stockage SPARQL et RDF	17
3.2.2.2 Mes considérations	18
3.2.3 Développer une plate-forme sémantique personnalisée en utilisant Flask ^[17] ou Django ^[18]	19
3.2.3.1 Flask, un microframework léger	19
3.2.3.2 Django, le framework web pour les perfectionnistes avec des délais	20
3.2.3.3 Mes considérations	20
3.3 Choix de la base de données	21
3.3.1 Bases de données RDF	21
3.3.2 Base de données Neo4j	22
3.3.2.1 Structure de Neo4j	22

3.3.2.2 Utilisation de Neo4j	22
3.3.3 Bases de données de graphes vs magasins triples RDF	22
3.3.3.1 Les similitudes entre les deux	23
3.3.3.2 Les différences entre les deux	23
3.3.4 neosemantics (n10s) ^[29] , boîte à outils Neo4j RDF et sémantique	24
3.3.5 Mes considérations	24
3.4 Bilan de mon stage	26
3.4.1 Obtenu des métadonnées via OAI-PMH	26
3.4.1.1 Entrepôt cible	26
3.4.1.2 Protocole OAI-PMH	27
3.4.1.3 Mon travail	27
3.4.2 Construction des ontologies via des fichiers XML	28
3.4.2.1 Owlready2, un package de programmation orienté ontologie en Python	28
3.4.2.2 Mon travail	28
3.4.3 Construction de web sémantique avec la base de données Neo4j et le framework Flask	30
3.4.3.1 Générer une base de données Neo4j avec une structure de type ontologie	30
3.4.3.2 Fonctions de base complètes des pages web via Flask	30
4 Conclusion	32

1 Introduction

Du 3 janvier 2022 au 3 juin 2022, j'ai eu la chance d'effectuer un stage au sein du Laboratoire ELLIADD (Édition, Littératures, Langages, Informatique, Arts, Didactiques, Discours) à distance et de travailler avec les membres du Projet HUMANE (HUMANités Numériques pour l'Éducation).

Au cours de ce stage et à travers ma mission du développement d'ontologie et de plate-forme sémantique, j'ai pu découvrir et apprendre le **Knowledge Graph (KG)**^[1] et le web sémantique, ainsi que comment utiliser les programmes Python pour obtenir des documents sous forme de XML, comment utiliser les programmes Python pour construire des ontologies, construire des bases de données Neo4j et des pages web Flask, et comment obtenir les métadonnées du service web via OAI-PMH^[2] via le programme Python. En conclusion, grâce à ce stage, j'ai acquis une compréhension de base et une expérience pratique des concepts et des opérations de l'ontologie, du Knowledge Graph et du web sémantique, ce qui a élargi mes perspectives de carrière. Non seulement j'ai saisi l'occasion d'apprendre à extraire du contenu pertinent via XML et d'autres types de fichiers, mais j'ai également appris à exploiter et à créer des bases de données Neo4j et à créer des pages web via Flask. J'ai fourni un soutien plus professionnel pour mon développement futur.

Le contenu de ma pratique cette fois est principalement divisé en deux parties, l'une est la composition de l'ontologie, et l'autre est la construction du web sémantique lié à l'ontologie. Dans une première partie, j'utiliserai le programme Python pour obtenir les informations dans les métadonnées du service web en extrayant la méthode OAI-PMH ou en extrayant les informations dans le fichier XML converti à partir de la carte mentale pour construire l'ontologie. Et une fois l'ontologie construite, elle est encore modifiée et enrichie. Dans la deuxième partie, la même méthode est utilisée pour obtenir des informations, et ces informations sont converties en base de données Neo4j selon le même format de l'ontologie précédente, et une plate-forme sémantique de base est construite via le framework Flask.

Ce rapport est organisé comme suit. Tout d'abord, je vais vous présenter l'institut, le contexte de la recherche connexe (se référant principalement au projet HUMANE), ainsi que le contexte et les objectifs de mon stage. Ensuite, je décrirai ma mission de stage et les raisons que j'ai choisies d'utiliser le framework Flask pour développer une plate-forme sémantique et d'utiliser Neo4j comme base de données. Enfin, je développerai quelques étapes importantes pour me présenter dans le développement avec le langage Python.

2 Monographie de l'entreprise

2.1 Présentation du laboratoire

L'EA 4661 ELLIADD (Édition, Littératures, Langages, Informatique, Arts, Didactiques, Discours) est une unité de recherche de l'Université de Franche-Comté reconnue par le Ministère et évaluée A par l'AERES pour son projet scientifique.

ELLIADD regroupe 74 enseignants-chercheurs, dont 23 HDR, une centaine de doctorants et 30 chercheurs associés, dans plusieurs champs disciplinaires associés : sciences du langage (CNU-7), langue et littérature françaises (CNU-9), sciences de l'information et de la communication (CNU-71), arts de la scène et musicologie (CNU-18), sciences et techniques des activités physiques et sportives (CNU-74) et sciences de l'éducation (CNU-70), mais aussi informatique, langues et littératures anglo-saxonnes, germaniques et slaves, psychologie, mécanique, histoire.

2.2 Recherches du laboratoire ELLIADD

Le laboratoire ELLIADD est issu de la fusion, en 2012, des laboratoires LASELDI et ATST-Centre Jacques-Petit, auxquels se sont joints d'autres chercheurs de l'UBFC autour d'un programme alliant approfondissement disciplinaire et synergie interdisciplinaire. Ainsi, l'unité de recherche est constituée de cinq pôles à l'identité forte et rattachée à de grands champs de la recherche liés à l'humain (analyse de discours, littérature, arts de la scène, archives et patrimoine, linguistique et sémiotique, didactique et FLE, sciences de l'information et de la communication, sciences de l'éducation, ergonomie et numérique). Chaque pôle incarne une coopération originale entre différentes disciplines du CNU et permet la mise en œuvre de projets transversaux fédérateurs. Deux programmes transversaux concernant les archives et les humanités numériques d'un côté, l'innovation pour l'éducation de l'autre, ont été constitués en axes structurants pour le laboratoire, au-delà de ses pôles.

2.2.1 Pôle « Arts et Littérature » (AL)

Les recherches du pôle « Arts et Littérature » (AL) concernent plus spécifiquement les arts de la scène, la musique et la littérature, avec un accent particulier mis sur la période allant du symbolisme à aujourd'hui. Fondé sur l'héritage du Centre Jacques-Petit, il a pour originalité de s'appuyer sur la gestion de fonds d'archives, en particulier celui consacré au poète, dramaturge et diplomate Paul Claudel.

2.2.2 Pôle « Conception, Crédit, Médiations » (CCM)

Les recherches du pôle « Conception, Crédit, Médiations » (CCM) ont l'ambition de contribuer à faire d'ELLIADD une référence majeure pour **les Sciences de l'Information et de la Communication (SIC)** notamment au regard du développement des **Humanités numériques**. Celles-ci introduisent un bouleversement épistémologique et sociétal aux enjeux techniques, esthétiques, éthiques, sociaux considérables. Dans une perspective latourienne, la médiation

ne se borne pas à relier, mais reconfigure et transforme ce qu'elle met en relation. Cette problématique de la médiation, centrale pour le pôle, permet d'aborder l'ère actuelle et future, avec à la fois une inscription forte de la communication dans le champ des sciences humaines et sociales, et des sciences de l'information et un dialogue disciplinaire renforcé notamment avec les sciences de l'éducation et les sciences du langage. C'est cette dynamique pluridisciplinaire qui permet en particulier d'aborder la complexité des dispositifs d'information et communication numérique analysés (en situation de formation et d'apprentissage, en situation de mobilité, en contexte professionnel, etc.).

Le pôle CCM interroge les processus communicationnels et artistiques, notamment autour des pratiques et usages des médias, de l'analyse de processus et de dispositifs communicationnels à la conception/réalisation de ces derniers. Les champs ou terrains d'applications vont de la communication électronique (des médias historiques aux médias sociaux), aux dispositifs de médiation (pédagogiques, culturelles), à l'organisation des connaissances, en s'appuyant sur différents outils théoriques ou techniques comme **le web sémantique**.

2.2.3 Pôle « Contextes, Langages, Didactiques » (CLD)

Les recherches conduites au sein du pôle « Contextes, Langages, Didactiques » (CLD) portent en particulier sur la diversité des langues, langages et cultures et de leurs conditions et modalités de médiation/circulation en relation avec les contextes éducatifs variés. La question de la transmission/appropriation des savoirs, à partir de situations plurimodales, plurilingues et multiculturelles et en contexte éducatif et/ou interculturel, positionne les objets d'étude autour des pratiques professionnelles comme phénomènes langagiers et faits de langue.

2.2.4 Pôle « Discours, Texte, Espace Public, Société » (DTEPS)

Le socle programmatique du pôle « Discours, Texte, Espace Public, Société » (DTEPS) est l'analyse du discours à forte consistance linguistique, textuelle, statistique et informatique, mais tournée vers de fortes coopérations interdisciplinaires, tout particulièrement avec les sciences de l'Information et de la communication, avec l'histoire et avec des ouvertures importantes vers l'anthropologie, la sociologie et la sociolinguistique.

2.2.5 Pôle « ERgonomie et COnception des Systèmes » (ERCOS)

Les recherches du pôle « ERgonomie et COnception des Systèmes » (ERCOS) associent sciences humaines et sociales, sciences de la vie et sciences pour l'ingénieur, en vue de développer des connaissances, des méthodes et des outils permettant une conception innovante de produit/système/service centrée sur l'homme, c'est-à-dire visant à préserver la santé, la sécurité et le bien-être, et à améliorer l'efficacité de la relation homme-machine en s'appuyant sur les compétences de l'utilisateur/opérateur.

2.3 Contexte de la recherche connexe

2.3.1 Formation du projet HUMANE

HUMANE (HUMANités Numériques pour l’Éducation) est un projet de recherche financé par le ministère de l’Éducation nationale, dans le cadre **des groupes thématiques numériques (GTnum)** de la mission d’incubation de la Direction Numérique pour l’Éducation, pour la période 2020-2022.

Le projet HUMANE s’inscrit dans l’axe 3 des GTnum : « **AXE 3. Humanités numériques, science et éducation ouverte : un nouveau défi pour la production et le partage des connaissances et des compétences au XXIe siècle** »^[3].

L’entité porteuse de ce projet est le **Groupement d’Intérêt Scientifique Innovation, Interdisciplinarité, Formation (GIS2if)**^[4]. Les co-animateurs scientifiques sont : Béatrice Drot-Delange et Thibaud Hulin.

2.3.2 Le but du projet HUMANE

Le projet HUMANE aborde le thème des humanités numériques comme un espace d’échange transversal, aux frontières de plusieurs disciplines (sciences, technologies, lettres et arts), favorisant une démarche d’apprentissage critique et réflexive.

Ce projet est destiné en premier lieu à recenser et de diffuser les propositions pédagogiques des praticiens et enseignants qui forment les élèves aux usages des outils numériques, à la créativité et à l’écriture numérique, ou qui initient les élèves à l’algorithme. La mise en commun a aussi pour but de clarifier les concepts abordés, de diffuser les pratiques susceptibles d’intéresser d’autres enseignants et de tenter d’en évaluer l’impact sur l’apprentissage, sur le projet professionnel des élèves plus âgés, et sur la société.

2.3.3 Coopération du projet HUMANE

Le projet HUMANE implique cinq équipes de recherche réparties sur les académies suivantes : Besançon, Clermont-Ferrand, Paris, Rennes, Versailles.

Les axes communs de travail sont les suivants :

- **décrire et partager les pratiques enseignantes** en éducation aux humanités numériques (scénarios pédagogiques et autres ressources) et ainsi favoriser le développement d’une didactique et d’une littératie numériques ;
- **cartographier** le domaine des humanités numériques pour l’éducation, en termes de compétences, spécifiques ou transversales (critiques et réflexives, créatives, communicationnelles) d’acteurs et de publications scientifiques ;
- **analyser la mise en œuvre d’enseignements** en humanités numériques, d’un point de vue disciplinaire et transversal, en lien avec le **contexte** social de l’élève (famille, relations sociales, citoyenneté) et avec son projet d’orientation **professionnelle** pour les plus âgés d’entre eux.

2.3.4 Intégration du projet HUMANE au sein du laboratoire ELLIADD

Rattaché au laboratoire ELLIADD, ce projet concerne tous ses pôles : la valorisation des corpus dans le champ de l'enseignement des HN mobilisera les pôles **AL**, **DTEPS** et **CLD** ; l'expérience en HN et en web sémantique mobilise le pôle **CCM** et sur son axe transversal SEISM portant sur la recherche en éducation ; le design de la plate-forme et son évaluation, le pôle **ERCOS**. 7 chercheurs ELLIADD travaillent déjà au projet HUMANE. Il s'agit donc d'un véritable programme de recherche structurant.

2.4 Contexte et objectifs du stage

2.4.1 Contexte du stage

Le déploiement des réseaux bouleverse les rapports aux savoirs et aux métiers d'enseignants et de chercheurs. Aux côtés d'agents intelligents capables de raisonner à partir d'ontologies (représentations structurées de domaines), les plates-formes sémantiques contribuent à établir cet espace commun de connaissances au XXI^e siècle : **les « Humanités Numériques » (HN)**.

Le thème des HN rassemble les travaux de chercheurs qui s'interrogent sur la nécessité de transmettre ces compétences cognitives fondamentales et avancées communes. Ce domaine doit également être mieux structuré, clarifié, motivé et partagé. Sa formalisation peut permettre d'étudier les liens entre concepts et compétences.

Au-delà, le thème de la transmission des HN a des implications pratiques, professionnelles et civiques importantes pour tous les jeunes et adultes en formation continue. L'identification des compétences avancées est stratégique pour promouvoir des compétences durables.

La recherche en intelligence artificielle (IA) s'est fortement développée ces dernières années (la France a investi 1,5 milliard d'euros en France en 2018). C'est un défi de construire un système d'intelligence artificielle pour stimuler les pratiques d'enseignement et de recherche en « **Humanités Numériques et Éducation** » (HNE), et de valoriser le corpus numérisé en tant qu'objet pédagogique du domaine de référence d'HNE.

2.4.2 Objectifs du stage

Grâce à l'avancement du projet HUMANE, les praticiens et chercheurs du projet ont lancé un corpus de ressources HNE (pédagogiques, projets, institutions ou personnes). Sur la base de l'établissement et de la mise à jour de ce corpus de ressources HNE, l'utilisation efficace de ce corpus est devenue un problème à explorer. L'objectif de ce stage est de développer l'ontologie et ses services intelligents à travers ce corpus HNE et les matériaux associés, et d'établir un web sémantique. Le but ultime du stage est de favoriser l'innovation dans l'enseignement, d'accroître la visibilité et d'améliorer la production et la recherche de corpus. En identifiant des compétences de haut niveau dans le domaine de l'ontologie, le projet vise à inspirer et à développer la pédagogie dans le domaine et à soutenir le lien entre la cognition et l'informatique. De plus, la plate-forme sémantique établie par le stage devrait également fournir des cas réalisables pour améliorer la recherche d'informations (à la fois, trouver des ressources, visualiser

des données et extraire de nouvelles connaissances). En conclusion, le but de ce stage est de construire le web Sémantique et d'envisager les services qu'il nous apportera dans le futur.

En raison de contraintes de temps, ce stage se concentre principalement sur la discussion, la conception et les méthodes pratiques au début du projet, en fournissant des solutions réalisables pour la mise en œuvre ultérieure du projet et en fournissant un éventuel soutien technique pour le projet connexe HUMANE.

3 Mes missions

3.1 Informations générales sur ce stage

3.1.1 Tâches à effectuer

En plus du contenu suivant qui a été refusé pendant le stage, les tâches principales de ce stage peuvent être divisées en quatre catégories suivantes :

- Convertir automatiquement des cartes mentales (il y a déjà la version v0.1) en une ontologie.
- Convertir automatiquement des textes collaboratifs en ligne en une ontologie.
- Sémantiser les données de bases d'expérience en innovation pédagogique (Édubase) pour les relier à l'ontologie et en permettre l'exploitation.
- Développer des services intelligents à partir de données collectées à l'aide d'une plate-forme sémantique comme Whyis ou Semantic Mediawiki.

Ces tâches sont en phase d'achèvement ou ont essentiellement établi une certaine base de faisabilité.

En fait, ces quatre catégories de tâches peuvent également être divisées en deux grandes parties, une partie est le développement de l'ontologie, et l'autre partie est le développement de la plate-forme sémantique, c'est-à-dire le développement de services intelligents liés à l'ontologie.

3.1.2 Modèle de stage

Mon stage a commencé le 3 janvier 2022 et a duré cinq mois jusqu'à la fin du 3 juin.

Au vu de l'épidémie covid-19 incertaine de ce début d'année et de certaines raisons particulières, l'ensemble du processus de ce stage a été accepté en stage à distance. En plus de communiquer par e-mails, mon tuteur de stage, Monsieur HULIN, et moi communiquons plus rapidement via le logiciel Discord^[5], en particulier pour les discussions sur certains problèmes clés. De plus, je m'assure également de soumettre chaque jour mon journal de travail de stage sur Discord afin que mon tuteur stagiaire puisse superviser mon contenu de travail quotidien.

En plus d'utiliser Discord pour communiquer, nous utilisons également la plate-forme Trello^[6] pour partager ou enregistrer des informations relatives au stage et planifier l'ensemble du processus de stage via cette plate-forme.

J'ai téléchargé l'intégralité de dossier des scripts du stage sur mon propre compte GitHub^[7]. À l'heure actuelle, j'ai signé un document de cession de droits avec mon tuteur stagiaire et l'ai défini comme un contenu de protection privée. Ce dossier sera trié et remis à mon tuteur de stage après la rédaction du rapport de stage.

Tous les scripts de ce stage sont écrits en langage Python, en utilisant la version Python 3.7 au lieu de la dernière version, Python 3.10, afin de pouvoir s'adapter à d'autres plates-formes de serveur d'hébergement web à l'avenir, telles que l'hébergement partagé fourni par A2 Hosting.

L'exécution de l'intégralité des scripts est effectuée dans l'environnement virtuel Python isolé créé par la bibliothèque à outil, virtualenv^[8]. Pendant toute la phase de mon stage, pour exécuter tous les scripts, les versions des bibliothèques principales utilisées sont montrées dans le Tableau 1.

Tableau 1 – Liste de la version des bibliothèques principales utilisées

```
beautifulsoup4==4.11.1
    # Bibliothèque pour récupérer des informations à partir de pages web.

Flask==2.1.1
    # Bibliothèques pour l'écriture de microframeworks web.

lxml==4.8.0
    # Bibliothèque facile à utiliser pour travailler avec XML et HTML.

Owlready2==0.37
    # Bibliothèques pour le développement de programmes orientés ontologie.

py2neo==2021.2.3
    # Bibliothèques pour manipuler les bases de données Neo4j.

spacy==3.3.1
    # Bibliothèques pour le traitement avancé du langage naturel.
```

3.2 Choix de la plate-forme sémantique

La tâche finale de ce stage est de réaliser la mise en place d'une plate-forme sémantique, cela implique donc la sélection d'une plate-forme sémantique. Au début du stage, mon tuteur de stage m'a proposé trois options à expérimenter et à pratiquer afin de faire le choix final. Permettez-moi de vous présenter ces plates-formes alternatives une par une.

3.2.1 Whyis^[9], un cadre de graphe de connaissances à l'échelle nanométrique

Whyis est un cadre de publication, de gestion et d'analyse de graphes de connaissances à l'échelle nanométrique. Ce projet est maintenu par tetherless-world. Whyis est conçu pour prendre en charge la gestion sensible au domaine et la gestion des connaissances provenant de nombreuses sources différentes. Son objectif principal est de pouvoir créer des graphes de connaissances utiles basés sur des domaines et des données. Les connaissances peuvent être apportées et gérées par l'interaction directe de l'utilisateur, l'analyse statistique ou l'extraction de données à partir de nombreux types de sources de données. Chaque contribution au graphe de connaissances est gérée comme une entité distincte afin que sa provenance (statut de publication, attribution et justification) soit transparente et puisse être gérée et utilisée.

Whyis gère ses connaissances comme des nanopublications, qui peuvent être considérées comme la plus petite unité publiable. Ce sont des fragments de graphes de connaissances avec des graphes auxiliaires qui leur sont associés pour contenir des informations de provenance et de publication. Un système de graphes de connaissances doit gérer la source de son

contenu. En utilisant les recommandations existantes telles que **RDF (Resource Description Framework)**, **W (Web Ontology Language)** et **SPARQL**, nanopublications est en mesure de fournir des options d'extension et d'intégration flexibles sans les limitations des outils de gestion de base de données personnalisés. Ils ont également la possibilité de capturer n'importe quel niveau de granularité d'informations requis par une application.

Chaque entité de la ressource est visible via son propre **URI (Uniform Resource Identifier)** et est disponible sous forme de données liées lisibles par machine. Lorsqu'un utilisateur visite un URI, toutes les nanopublications le concernant sont agrégées dans un graphe. Cette approche permet aux utilisateurs de contrôler l'accès à ces connaissances. Il offre également la possibilité de contrôler le flux de travail de publication. Au lieu de tout publier en même temps, NanoPublications peut être contribué, organisé et approuvé, puis éventuellement publié individuellement ou en tant que collection. Les développeurs de graphes de connaissances peuvent contrôler de manière flexible la façon dont les entités sont affichées aux utilisateurs via leur type ou d'autres contraintes. Whyis fournit des vues par défaut pour la création de graphes de connaissances, y compris le développement d'ontologies, et permet également aux développeurs de fournir des vues personnalisées.

3.2.1.1 Graphique des connaissances à l'échelle nanométrique

Le graphique des connaissances à l'échelle nanométrique est construit à partir d'un certain nombre de nanopublications, dont chacune est suivie individuellement, permettant une justification basée sur l'attribution et un crédit de publication pour chaque élément de connaissance dans le graphique.

Les nanopublications sont des graphes RDF qui encodent des faits scientifiques extraits de la littérature, enrichis d'informations de provenance et d'attribution^[10]. Il existe actuellement des millions de nanopublications sur le web, notamment dans les sciences de la vie. Les nanopublications sont considérées comme facilitant la découverte, l'exploration et la réutilisation des faits scientifiques. Pourtant, ils ne sont pas largement utilisés par les scientifiques en dehors de certains cercles. Ils sont difficiles à trouver et rarement cités. Ceci est généralement considéré comme étant dû à un manque de services pour trouver, trouver et comprendre le contenu des nanopublications.

Dans Whyis, l'utilisateur n'est pas obligé de tenter de minimiser une unité de pensée. Si un ensemble d'assertions, c'est-à-dire la plus petite unité d'informations publiées, a la même provenance et les mêmes informations publiées, elles peuvent être regroupées. Par exemple, si un graphe de connaissances doit référencer une ontologie, cela peut être fait dans une nanopublication, même si elle peut être très volumineuse. Si la provenance de ces assertions est commune, il est raisonnable de les conserver comme une seule assertion. Cependant, si les assertions pouvaient être décomposées en unités plus petites, cela faciliterait leur gestion et leur version.

3.2.1.2 Architecture de Whyis

Whyis est écrit en Python à l'aide du framework Flask et utilise de nombreux outils d'infrastructure existants pour fonctionner (Figure 1). La base de données RDF utilisée par défaut est **Blazegraph™^[11]**, qui fournit une mise à l'échelle horizontale pour les grands graphiques. Whyis utilise le protocole HTTP SPARQL 1.1 de requête, de mise à jour et de stockage de graphes. Toute base de données RDF prenant en charge ces protocoles peut remplacer Blazegraph.

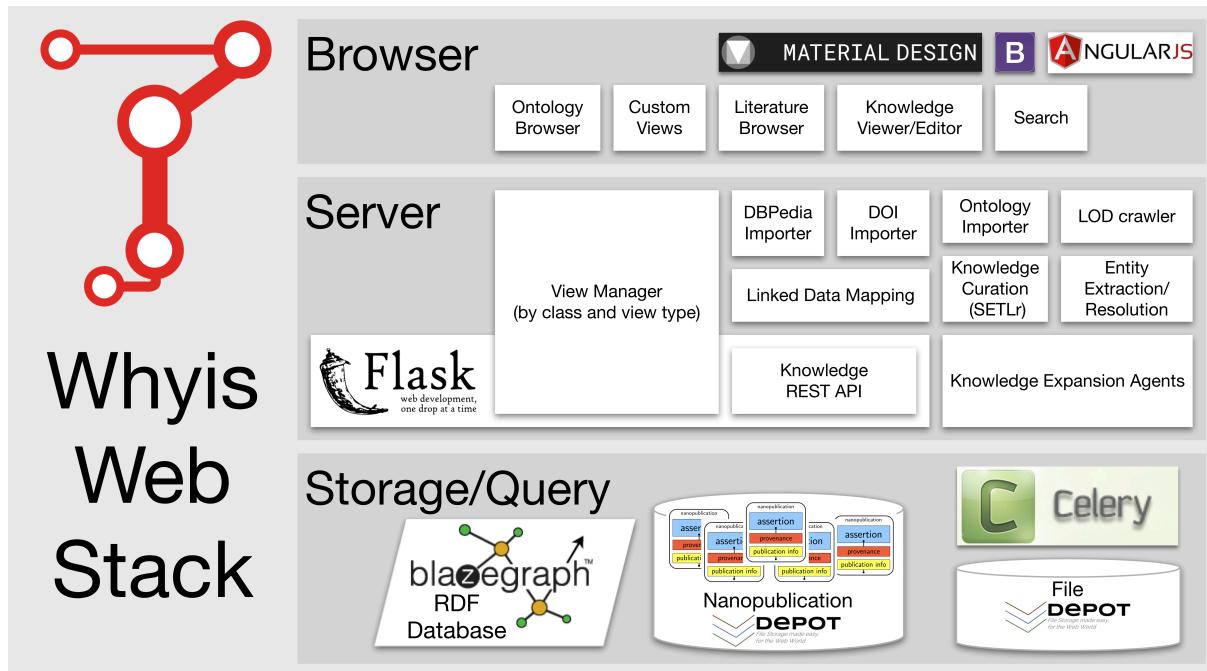


Figure 1 – Architecture de Whyis

Whyis fournit un stockage à l'aide de la bibliothèque **FileDepot Python^[12]** pour fournir des nanopublications basées sur des fichiers et la persistance des fichiers téléchargés. FileDepot résume la couche de stockage dans un backend configurable qui gère le stockage des types de contenu, des noms de fichiers et d'autres métadonnées, et fournit un identifiant persistant pour chaque fichier. Il fournit des backends pour plusieurs méthodes de stockage de fichiers, y compris les fichiers locaux, Amazon S3, MongoDB GridFS et les bases de données relationnelles.

Whyis s'appuie également sur un système de file d'attente de tâches appelé **Celery^[13]**, qui peut être mis à l'échelle en ajoutant plus de travailleurs de tâches sur des machines distantes. Pour prendre en charge cela, Whyis peut être configuré comme un système distribué sans partage en configurant File Depot pour utiliser l'un des Amazon S3, GridFS ou l'un des backends de stockage de base de données.

En bref, Whyis stocke la nanopublication actuelle dans une base de données RDF, tandis que tout l'historique est stocké dans l'archive de fichiers de nanopublication à l'aide de File Depot. Les fichiers peuvent également être téléchargés et stockés dans une instance spéciale de File Depot. Celery est utilisé pour invoquer et gérer un ensemble d'agents d'inférence autonomes qui écoutent les changements de graphes et répondent aux nanopublications supplémentaires.

Les utilisateurs interagissent avec le graphique via un ensemble de vues configurées par type de nœud et basées sur le système de modèles de **Flask Ninja2**.

3.2.1.3 Mes considérations

Alors que le framework Whyis est une solution pour créer des plates-formes de renseignement spécifiques à HNE. Il peut extraire la plus petite unité d'information, la nanopublication, d'une ressource à des fins d'inférence. Il sait comment gérer des sources de données hétérogènes ainsi que le langage naturel, manipulant des graphes de connaissances publiques pour répondre aux questions des enseignants et des chercheurs. En outre, Whyis a accumulé une expérience dans le traitement des nanopolymères, la politique du spectre et en particulier l'informatique de la santé^[14].

Mais c'est précisément à cause du cadre complexe et serré de Whyis que cela augmente vraiment la difficulté de démarrer pour un débutant de cette application. De la récupération et du stockage des nanopublications au débogage et à la gestion du serveur, tous ont créé d'énormes défis pour les débutants. Bien que Whyis soit un cadre de programmation en Python, l'éventail des connaissances impliquées dans son application est très large et relativement peu connu, et il nécessite un soutien technique considérable ou des discussions communautaires tout au long du processus pour assurer le bon déroulement de l'ensemble du processus.

Limité par la façon dont je pratique à distance, bien que Whyis ait été installé avec succès sur le serveur par la **Délégation Régionale du Numérique pour l'Éducation de la région académique Bourgogne-Franche-Comté (DRNE)** il y a longtemps. Cependant, pour diverses raisons compliquées telles que les versions alternées des différentes bibliothèques utilisées par Whyis, je n'ai pas réussi à l'installer moi-même dans le système virtuel de mon ordinateur. De plus, comme ce cadre est dominé par le tiers, l'Institut polytechnique Rensselaer, tout le support technique nécessite une assistance à distance de ce tiers. En revanche, la propre page d'accueil de Whyis manque de support technique détaillé et de listes de cas, ce qui pose de nombreux défis pour traiter divers petits problèmes encombrants. Après tout, la mise en place de cette plate-forme n'est que la dernière partie de mon stage, et j'ai également besoin de passer beaucoup de temps sur le développement de l'ontologie précoce. Par conséquent, bien que je pense que Whyis semble être un cadre très complet et précieux, en raison de contraintes de temps et d'obstacles de support technique, je pense que ce n'est peut-être pas le meilleur choix pour ma tâche actuelle.

3.2.2 Semantic MediaWiki^[15]

Semantic MediaWiki (SMW) est une extension libre et d'open source du MediaWiki^[16] (Figure 2) qui alimente Wikipédia, permettant aux utilisateurs de stocker et d'interroger des données dans les pages wiki.

Dans le même temps, Semantic MediaWiki est un cadre à part entière qui, combiné à de nombreuses extensions dérivées, peut transformer un wiki en un système de gestion des connaissances puissant et flexible. Toutes les données créées dans SMW peuvent être facilement pu-

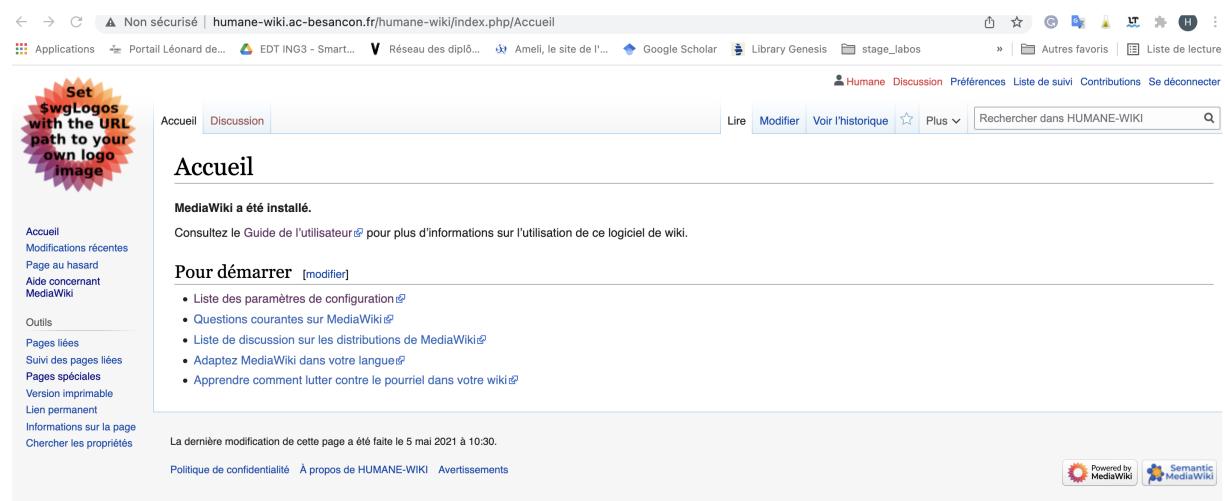


Figure 2 – MediaWiki avec l'extension Semantic MediaWiki

bliées sur le web sémantique, permettant à d'autres systèmes d'utiliser les données de manière transparente.

3.2.2.1 SMW utilise le stockage SPARQL et RDF

Les magasins RDF sont parfois appelés « magasins triples », bien que de nombreux magasins modernes soient en fait des « magasins quadruples », qui attribuent également un graphe nommé à chaque triplet RDF.

Par défaut, SMW stocke toutes les données dans la même base de données relationnelle (généralement une base de données MySQL) utilisée par MediaWiki. Cela garantit une configuration simple, mais les bases de données relationnelles ne sont pas le type de stockage idéal pour les données sémantiques. Un modèle de données plus naturel pour les données SMW est RDF, un format de données qui organise les informations dans des graphiques plutôt que dans des tables de base de données fixes. Heureusement, un système basé sur RDF peut être utilisé conjointement avec une base de données SQL standard pour gérer et interroger les données de SMW.

- **Avantages d'utiliser une base de données RDF**

- Meilleures performances des requêtes Les magasins RDF sont conçus pour répondre aux requêtes dans le langage de requête SPARQL. Les requêtes SMW peuvent être exprimées dans ce langage beaucoup plus naturellement que dans le langage de requête SQL des bases de données relationnelles. En ce sens, les requêtes SMW sont un cas d'utilisation principalement typique pour les systèmes de bases de données RDF alors qu'elles sont un cas d'utilisation plutôt exotique pour les systèmes de bases de données relationnelles.
- Interfaces supplémentaires Les magasins RDF qui prennent en charge la norme SPARQL permettent également à d'autres applications de poser des requêtes SPARQL sur leurs données sans passer par l'interface web SMW. Cela permet une utilisation efficace des données wiki dans d'autres applications.

- Fonctionnalités de raisonnement et accès aux données basé sur l'ontologie Les langages du web sémantique tels que RDF Schema et OWL fournissent des fonctionnalités expressives supplémentaires pour la modélisation, par exemple en permettant la déclaration de classes dérivées ou la déclaration d'autres caractéristiques de propriété (par exemple, la transitivité des propriétés). Certaines bases de données compatibles SPARQL peuvent évaluer ces fonctionnalités pour répondre aux requêtes.
- Intégration de données et réutilisation d'ontologies Il est possible de stocker des données supplémentaires dans la base de données RDF qui est mise à jour par SMW. De cette manière, le magasin RDF peut agir comme une plate-forme pour l'intégration de données et la réutilisation d'ontologies.
- Séparation physique des ressources informatiques L'utilisation d'une base de données différente de celle de MediaWiki offre un moyen simple de répartir les tâches sur plusieurs serveurs. En particulier, les requêtes complexes peuvent ainsi être empêchées d'affecter le fonctionnement de base du wiki, même si elles consomment de manière inattendue une puissance de calcul prohibitive, c'est-à-dire si elles tuent le serveur qui héberge la base de données RDF.

- **Inconvénients de l'utilisation d'une base de données RDF**

- Exigences de stockage plus élevées Les données sont uniquement reflétées dans les bases de données RDF, pas supprimées de SQL. Un espace de stockage supplémentaire est donc nécessaire.
- Effort de maintenance supplémentaire La configuration des backends RDF dans SMW est facile, mais il reste encore quelques efforts à faire pour exécuter un système de gestion de base de données supplémentaire.
- Questions concernant les performances et la stabilité Il existe aujourd'hui un certain nombre de bases de données RDF de pointe, dont certaines sont gratuites/open source. Cependant, l'expérience de l'utilisation de ces systèmes avec SMW est encore limitée, donc certains tests sont utiles avant de décider d'un backend particulier pour une application SMW à grande échelle.

3.2.2.2 Mes considérations

Le choix de la plate-forme sémantique s'est fait au début de mon stage, lorsque la version grand public de Semantic MediaWiki était encore SMW 3.2.3, donc la version testée à cette époque était SMW 3.2.3. Il n'a pas fallu longtemps à SMW pour lancer sa dernière version majeure, SMW 4.0.0, le 18 janvier. Ensuite, SMW a lancé SMW 4.0.1 le 24 mars, la première version après la version majeure de SMW 4.0.0. Il s'agit d'une version de maintenance qui fournit des corrections de bogues et des mises à jour de traduction.

Bien que l'installation de SMW ne soit pas un gros défi, et qu'il semble que SMW rassemble la plus grande communauté, il y a peu de discussions sur des cas tels que l'importation d'ontologies, et le manque de mises à jour rend plus difficile l'utilisation de SMW.

De plus, SMW fournit des plug-ins tiers pour l'importation d'ontologies, mais certains de ces plug-ins sont sérieusement en retard dans la mise à jour et ne sont plus applicables à l'environ-

nement système actuel et ne peuvent pas être installés. Cependant, d'autres plug-ins ont des fonctions très simples, manquent de fonctions d'interface utilisateur riches et ne prennent pas en charge les fonctions de gestion et d'édition du contenu de l'ontologie. Tous ces problèmes rendent mon expérience avec SMW pas idéale.

Considérant que les performances et la stabilité de l'utilisation ultérieure de la base de données RDF par SMW peuvent être un défi potentiel à l'avenir, je pense que ce n'est pas nécessairement un bon choix d'utiliser SMW comme plate-forme sémantique à ce stade.

3.2.3 Développer une plate-forme sémantique personnalisée en utilisant Flask^[17] ou Django^[18]

3.2.3.1 Flask, un microframework léger

Flask est un **microframework**^[19] léger basé sur Python disponible sous la licence **BSD (Berkeley Source Distribution)**^[20] pour développer des applications web en Python. Flask est plus simple et plus léger à utiliser que les autres frameworks web basés sur Python, mais inclut toutes les fonctionnalités dont vous avez besoin pour développer des applications petites, mais puissantes. Flask est extensible et offre une variété d'extensions, ce qui facilite davantage le développement d'applications robustes.

Flask est appelé un « microframework », non pas parce que Flask manque de fonctionnalités, mais parce qu'il garde le noyau simple, mais extensible. Par défaut, Flask n'inclut pas de couche d'abstraction de base de données, de validation de formulaire ou de tout autre composant fourni par des fonctionnalités communes à partir de bibliothèques tierces préexistantes. Au lieu de cela, Flask prend en charge les extensions pour ajouter de telles fonctionnalités à l'application de l'utilisateur comme si elle était implémentée dans Flask lui-même. De nombreuses extensions fournissent l'intégration de bases de données, la validation de formulaires, la gestion des téléchargements, diverses technologies d'authentification ouvertes, etc. Le Flask est peut-être « petit », mais il est prêt à être utilisé en production pour une variété de besoins. En conclusion, Flask ne prend pas beaucoup de décisions pour l'utilisateur, comme la base de données à utiliser. Les décisions qu'il prend, telles que le moteur de modèle à utiliser, sont faciles à modifier. Tout le reste dépend de l'utilisateur, donc Flask peut être tout ce dont l'utilisateur a besoin, pas tout ce dont l'utilisateur n'a pas besoin.

Flask dépend du moteur de template **Jinja**^[21] et de la boîte à outils **Werkzeug**^[22] **WSGI (web Server Gateway Interface)**^[23].

Voici quelques-unes des caractéristiques notables de Flask :

- Serveur de développement intégré pour l'hébergement d'applications web et le débogueur.
- Prise en charge intégrée des tests unitaires.
- Style RESTful de répartition des requêtes.
- Modèle Jinja2.
- Prise en charge des cookies sécurisés (sessions côté client).

- 100 % conforme à WSGI 1.0.
- Compatible avec Google App Engine.
- Documentation complète et mise à disposition d'une large gamme d'extensions^[24].

3.2.3.2 Django, le framework web pour les perfectionnistes avec des délais

Les frameworks web ou les frameworks d'applications web sont le moyen standard, structuré et plus rapide de développer des applications web qui peuvent être exposées sur Internet à une grande variété d'utilisateurs.

Django est l'un des puissants frameworks web écrits en Python pour créer rapidement des applications web basées sur Python en quelques minutes. Il s'agit d'un framework web Python de haut niveau qui encourage un développement rapide et une conception propre et pragmatique. Construit par des développeurs expérimentés, il prend en charge une grande partie des tracas du développement web, afin que les développeurs puissent se concentrer sur l'écriture de leur application sans avoir à réinventer la roue. En dehors de cela, Django est un framework gratuit et open source.

Bien qu'il existe de nombreux autres frameworks web Python, Django est l'un des choix évidents pour créer des systèmes d'entreprise et à grande échelle. Basé sur le modèle **MVC (Model-View-Controller)**^[25], Django est rapide, sécurisé et évolutif.

La distribution source Python a longtemps maintenu **la philosophie des « piles incluses »**^[26] — avoir une bibliothèque standard riche et polyvalente qui est immédiatement disponible, sans obliger l'utilisateur à télécharger des packages séparés. Cela donne au langage Python une longueur d'avance dans de nombreux projets.

Django suit la philosophie des « piles incluses » de Python, où il fournit divers outils et utilitaires tels que la création de modèles, les formulaires, le routage, l'authentification, l'administration de base de données de base, etc., ce qui permet aux développeurs de plonger et de développer facilement des applications web dans les plus brefs délais.

3.2.3.3 Mes considérations

En général, cette troisième option me demande de développer une plate-forme personnalisée avec des outils Python et de construire une interface utilisateur simple avec Django ou Flask. C'est sans aucun doute un grand défi, car tout doit être fait à partir de zéro.

Cependant, d'un autre point de vue, des défis plus importants signifient souvent des gains plus importants. Par rapport aux deux premières options, bien que la construction d'une plate-forme personnalisée à partir de zéro nécessite beaucoup plus d'éléments que les deux premières, en particulier dans la conception initiale du projet, de nombreuses connaissances préalables doivent être acquises. Bien que j'aie une certaine expérience dans le développement React Native et Angular, et que je sois très habitué aux projets de programmation basés sur le langage Python, je suis encore nouveau dans le développement web avec Django ou Flask. Cependant,

si j'ai la chance de maîtriser l'un de ces deux frameworks de développement web lors de mon stage, ce sera un gros gain pour moi, et il sera probablement utilisé dans mes futurs travaux.

Par rapport aux deux premières options, bien que cette troisième option semblera plus vague et vide au début du projet, elle est sans aucun doute plus difficile, avec un espace d'expansion plus flexible et plus de possibilités potentielles à l'avenir. Par rapport aux deux premières options, cette troisième option peut réservier plus d'espace de personnalisation aux utilisateurs ou développeurs à l'avenir, qu'il s'agisse de la sélection de la base de données correspondante dans le futur, ou de l'ajout et de l'amélioration de fonctions personnalisées ou même de l'interface utilisateur. L'optimisation et l'amélioration peuvent être ajoutées ou améliorées une par une dans les travaux futurs. Dans une certaine mesure, cela contribuera sans aucun doute à prolonger la ductilité et la longévité de l'ensemble du projet.

En fait, pour le dire crûment, cette troisième option est une décomposition de la première option — Whyis. Après tout, Whyis est aussi une application écrite à l'aide du framework Flask. Mon travail consiste à implémenter une plate-forme sémantique similaire à Whyis par d'autres moyens. Parmi eux, la plus grande différence que j'utilise est le choix de la base de données, et je le présenterai en détail dans la section 3.3.

3.3 Choix de la base de données

Le choix de la base de données est peut-être la partie la plus controversée de mon stage. Comme le programme Whyis que j'ai présenté dans la section 3.2 utilise Blazegraph™ DB dans son architecture, et j'utilise la base de données Neo4j^[27] dans ce projet de stage, les deux sont des bases de données de graphes, mais le premier a des API RDF/SPARQL, le second n'en a pas. Permettez-moi d'expliquer mes choix à l'époque en présentant et en comparant ces types de bases de données.

3.3.1 Bases de données RDF

Les bases de données RDF, également connues sous le nom de « magasins triples », sont un type de base de données de graphes qui stocke des triplets. Les triplets sont des points de données représentés dans une relation sujet-prédicat-objet. Ces bases de données effectuent une récupération dans ce triple format comme un moyen plus spécifique de recevoir et de visualiser les données. Les bases de données RDF sont avantageuses, car elles organisent les informations dans ces ensembles de triplets, mais elles peuvent également être affichées à l'utilisateur sous forme de graphiques. Les données peuvent être extraites de bases de données RDF avec une variété de langages de requête, mais le plus courant est SPARQL.

Pour pouvoir être inclus dans la catégorie Base de données RDF, un produit doit^[28] :

- fournir un stockage de données ;
- enregistrer les données sous forme de triplets ;
- autoriser les utilisateurs à récupérer les données à l'aide du langage de requête.

Blazegraph™ DB est une base de données de graphes ultra performante prenant en charge

les Blueprints et les API RDF/SPARQL. Il prend en charge jusqu'à 50 milliards d'arêtes sur une seule machine. Il est utilisé en production pour les clients Fortune 500 tels qu'EMC, Autodesk et bien d'autres. Il prend en charge les applications clés de la médecine de précision et est largement utilisé pour les applications des sciences de la vie. Il est largement utilisé pour prendre en charge la cyberanalyse dans les applications commerciales et gouvernementales. Il alimente le service de requête Wikidata de la Wikimedia Foundation.

3.3.2 Base de données Neo4j

Neo4j est le système de gestion de base de données le plus connu et c'est aussi un système de base de données NoSQL. Neo4j est différent de Mysql ou MongoDB, il a ses propres fonctionnalités qui le rendent spécial par rapport aux autres systèmes de gestion de base de données.

3.3.2.1 Structure de Neo4j

Neo4j stocke et présente les données sous forme de graphique non au format tabulaire ou non au format JSON. Ici, toutes les données sont représentées par des nœuds et là, l'utilisateur peut créer une relation entre les nœuds. Cela signifie que l'ensemble de la collection de bases de données ressemblera à un graphique, c'est pourquoi cela la rend unique par rapport aux autres systèmes de gestion de bases de données. MS Access, serveur SQL, tous les systèmes de gestion de bases de données relationnelles utilisent des tables pour stocker ou présenter les données à l'aide de colonnes et de lignes, mais Neo4j n'utilise pas de tables, de lignes ou de colonnes comme le style « old school » pour stocker ou présenter les données.

3.3.2.2 Utilisation de Neo4j

Si le système de gestion de base de données a autant de relations d'interconnexion, l'utilisateur pourra utiliser Neo4j qui sera le meilleur choix. Neo4j est hautement préférable pour stocker des données contenant plusieurs connexions entre les nœuds. C'est là que Neo4j (bases de données de graphes) entre en jeu, il est plus confortable à utiliser avec des données relationnelles qu'avec la base de données relationnelle. Parce que Neo4j ne nécessite pas de schéma prédéfini, il suffit de charger les données ici, les données sont la structure.

Certaines fonctionnalités uniques feront l'utilisateur qui choisit Neo4j par rapport à tout autre système de gestion de base de données. Neo4j est entouré de relations, mais il n'est pas nécessaire de configurer des contraintes de clé primaire ou de clé étrangère pour les données. Ici, l'utilisateur pourra ajouter n'importe quelle relation entre les nœuds de son choix. Cela rend le Neo4j extrêmement adapté aux données de mise en réseau.

3.3.3 Bases de données de graphes vs magasins triples RDF

La principale différence entre les bases de données de graphes et les magasins triples est la façon dont ils modélisent le graphe. Dans un magasin triple (ou magasin quadruple), les données ont tendance à être très atomiques. Cela signifie que les « nœuds » dans le graphe

ont tendance à être des types de données primaires comme une chaîne, un entier, une date, etc. ou une relation, généralement.

En revanche, d'autres bases de données de graphes sont souvent appelées « magasins de propriétés », car les nœuds sont des conteneurs de données qui correspondent aux objets d'un domaine. Un nœud remplace un objet et possède des propriétés ; ils agissent comme des types de données riches spécifiés par les modélisateurs de graphes, plus que de simples types de données primaires. Dans ces bases de données de graphes, les nœuds et les relations sont « l'unité de discours ».

Notez qu'en RDF, il y a 3 relations, mais qu'une seule de ces relations exprime réellement la sémantique entre deux entités. Les deux autres relations ne font que suivre les propriétés d'une seule entité de niveau supérieur. Dans Neo4j, il s'agit d'une relation entre deux nœuds, chaque nœud ayant une propriété. En RDF, l'utilisateur aura tendance à identifier les choses par URI, dans neo4j c'est un objet de base de données qui obtient automatiquement un ID de base de données. Cela signifie à peu près la différence entre un magasin plus atomique / primaire (magasins triples) et un graphe de propriétés plus riche.

Pour résumer, les bases de données de graphes et les magasins triples sont conçus pour stocker des données liées. RDF est un type spécifique de données liées qui sont interrogées à l'aide de SPARQL, il est donc juste de dire que les magasins triples RDF sont une sorte de base de données de graphes. Cependant, il existe des différences subtiles, mais importantes qui sont décrites ci-dessous.

3.3.3.1 Les similitudes entre les deux

- Les bases de données de graphes et les magasins triples RDF se concentrent sur les relations entre les données, souvent appelées « données liées ». Les points de données sont appelés nœuds et la relation entre un point de données et un autre est appelée arête.
- Un réseau de nœuds et d'arêtes peut être assemblé dans des visualisations intéressantes, une caractéristique déterminante des bases de données de graphes.

3.3.3.2 Les différences entre les deux

- Les bases de données de graphes sont plus polyvalentes avec les langages de requête : Neo4j peut exécuter un magasin triple RDF et utiliser SPARQL, mais se concentre généralement sur son propre langage propriétaire, Cypher. Les magasins triples RDF utilisent uniquement SPARQL comme langage de requête.
- Les bases de données de graphes peuvent stocker divers types de graphes, y compris des graphes non orientés, des graphes pondérés, des hypergraphes, etc. Les magasins de triplets RDF se concentrent uniquement sur le stockage des rangées de triplets RDF.
- Les bases de données de graphes sont centrées sur les nœuds ou les propriétés, tandis que les magasins triples RDF sont centrés sur les bords. Les magasins triples RDF ne sont en réalité qu'une liste d'arêtes de graphe, dont beaucoup sont des « propriétés » d'un nœud et ne sont pas critiques pour la structure du graphe elle-même.

- Les bases de données de graphes peuvent stocker divers types de graphes, y compris des graphes non orientés, des graphes pondérés, des hypergraphes, etc. Les magasins de triplets RDF se concentrent uniquement sur le stockage des rangées de triplets RDF. Il s'agit d'une différence fondamentale entre les magasins triples RDF et les bases de données de graphes.
- Les magasins triples RDF fournissent également des inférences sur les données, mais pas les bases de données de graphes (par exemple, si les humains sont une sous-classe de mammifères et que l'homme est une sous-classe d'humains, alors on peut en déduire que l'homme est une sous-classe de mammifères).
- Les magasins triples RDF sont plus synonymes de « web sémantique » et de l'univers standardisé des connaissances stockées sous forme de triplets RDF sur DBpedia et d'autres sources, tandis que les bases de données de graphes sont considérées comme plus pragmatiques qu'académiques.

3.3.4 neosemantics (n10s)^[29], boîte à outils Neo4j RDF et sémantique

Ces dernières années, le système de base de données de graphes représenté par Neo4j s'est développé rapidement. Afin de permettre l'utilisation de RDF, modèle standard du W3C pour l'échange de données, et de ses vocabulaires associés dans Neo4j, Neo4j a lancé un plug-in spécial appelé neosemantics (n10s). N10s est un plug-in qui permet l'utilisation de RDF et de ses vocabulaires associés comme OWL, RDFS, SKOS (Simple Knowledge Organization System) et autres dans Neo4j. En plus d'utiliser n10s pour créer des intégrations avec des composants producteurs/consommateurs RDF, les utilisateurs peuvent également l'utiliser pour valider le graphique d'un utilisateur ou exécuter une inférence de base par rapport aux contraintes exprimées dans SHACL.

Ses fonctionnalités comprennent :

- import ou export de RDF et RDF* dans plusieurs formats (Turtle, N-Triples, JSON-LD, RDF/XML, TriG et N-Quads, Turtle*, TriG*);
- mappage du modèle à l'import ou l'export;
- importation et exportation d'ontologies/taxonomies dans différents vocabulaires (OWL, SKOS, RDFS);
- validation de graphe basée sur les contraintes SHACL ;
- inférence de base.

3.3.5 Mes considérations

Les magasins triples RDF sont principalement conçus pour les types de défis architecturaux rencontrés par les utilisateurs avec le web sémantique.

Les bases de données de graphes, en revanche, sont adaptées à différents cas d'utilisation. L'utilisateur aura tendance à identifier les choses avec son propre schéma.

Neo4j est une solution mature qui popularise les graphiques de propriétés et les rend faciles à utiliser. Il n'y a pas non plus de réponse définitive à la question de savoir s'il peut être pleinement

utilisé pour le travail sur le web sémantique. Il est généralement admis que, bien que les graphes ne soient pas des ontologies, les ontologies peuvent être représentées dans des graphes, et un graphe RDF en est un exemple.

Neo4j, en tant que base de données de graphes bien connue, est livré avec un langage de requête supplémentaire pour les explorer, qui est Cypher. Bien que Cypher et SPARQL aient des syntaxes complètement différentes, ils ont à peu près les mêmes fonctionnalités^[30]. Bien sûr, au-delà de la syntaxe, la principale différence fonctionnelle entre SPARQL et Cypher provient de la différence entre **Linked Property Graph (LPG)** et le modèle de données de graphe RDF standard^[31]. LPG autorise les propriétés de bord, contrairement au RDF standard. De sorte que les deux ont des capacités différentes dans la représentation des informations de schéma et le raisonnement graphique^[32]. Mais LPG et RDF ont une chose en commun, les deux traitent les données comme des graphiques. Sans surprise, il existe des moyens de prouver qu'il est possible de convertir un format à un autre^[33].

La traduction de SPARQL en Cypher est un processus complexe, car il existe des différences à la fois dans la syntaxe du langage et dans la manière dont les données sont stockées sur les plates-formes concernées, ce qui rend très difficile la corrélation complète entre les deux. Les recherches existantes nous ont montré une partie de ce processus, depuis l'explication des moteurs fonctionnant dans ces langues jusqu'aux détails de chaque langue, et la démonstration des capacités de traduction des parseurs mis en œuvre jusqu'à présent, ainsi que la démonstration qu'une partie du processus de validation est une grammaire bien développée^[34].

La raison pour laquelle j'ai choisi Neo4j au départ était principalement de considérer que ces dernières années, en raison du développement rapide des systèmes de bases de données de graphes représentés par Neo4j, l'utilisation de bases de données de graphes pour gérer les triplets RDF est également devenue l'un des choix. Étant donné que la plupart des bases de données de graphes ne peuvent pas directement prendre en charge le triple stockage RDF, dans ce cas, une méthode de conversion de données peut être utilisée pour prétraiter RDF dans un format de données pris en charge par les bases de données de graphes (comme les modèles de graphiques de propriétés), puis effectuer une gestion ultérieure. Neo4j fournit aux utilisateurs des plug-ins pour importer RDF^[35] ou ontologie^[36], bien qu'il y ait encore quelques lacunes dans la fonction. Sur la base des attentes de Neo4j, et du test et de l'expérimentation de son application au web sémantique, j'ai choisi d'utiliser Neo4j en stage.

Pour cette tentative, le plus gros défi est que Neo4j natif n'a pas la capacité de raisonner sur les données. Cependant, un article souligne qu'il existe une technologie appelée GraphScale^[37] qui fournit une inférence OWL évolutive pour Neo4j^[38]. La combinaison de la technologie Neo4j et GraphScale a un grand potentiel. Si Neo4j peut simuler avec succès la capacité de raisonnement de données similaire à RDF avec cette technologie, cela ne fera qu'ajouter un coup de pouce à ce projet à l'avenir.

3.4 Bilan de mon stage

Pendant tout le stage, j'ai complété les scripts ou programmes suivants (Figure 3). J'ai marqué les progrès respectivement. Maintenant, permettez-moi de vous présenter quelques-uns des scripts ou programmes les plus importants.

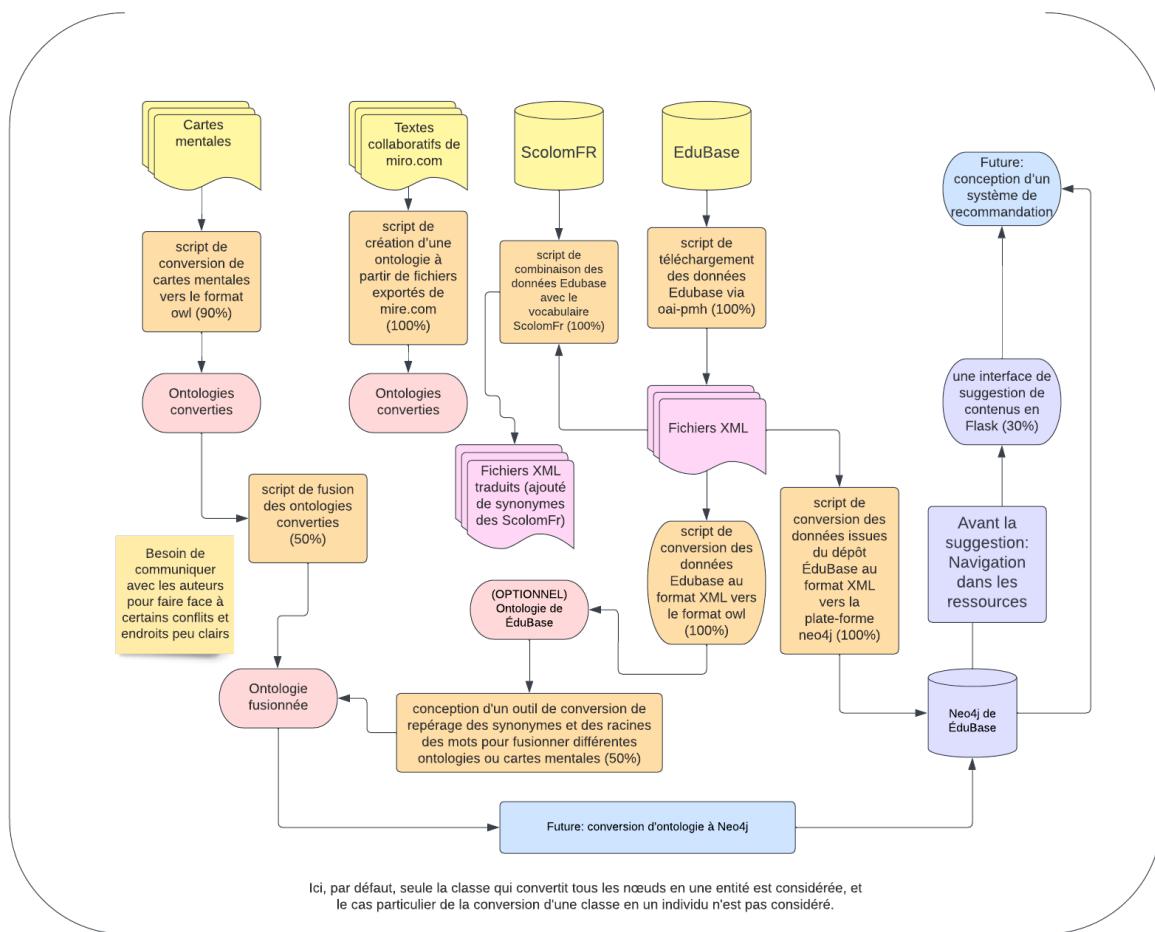


Figure 3 – Avancement du stage

3.4.1 Obtenu des métadonnées via OAI-PMH

3.4.1.1 Entrepôt cible

Édubase^[39] est une banque nationale de scénarios pédagogiques opérée par la Direction du Numérique pour l'Éducation (DNE). Elle permet, à partir d'une interface unique, de rechercher un scénario pédagogique élaboré en académie illustrant un thème de programme en lien avec le numérique éducatif. Les scénarios qui y sont indexés couvrent toutes les disciplines, tous les enseignements et tous les niveaux (premier et second degré, enseignement supérieur).

Les fiches présentes dans Édubase décrivent des scénarios adaptables et exploitables, créés par des enseignants, validés par les corps d'inspection, publiés en académie puis indexés dans la base nationale. Ces fiches pointent directement vers les pages des sites académiques où sont publiés les scénarios. L'entrepôt OAI-PMH d'Édubase contient l'ensemble des fiches en

XML (Figure 4).

The screenshot shows a web browser window with the title "OAI 2.0 Request Results". The URL is https://edubase.eduscol.education.fr/entrepot/oaipmh?verb=ListMetadataFormats. The page content is an XML document with the following structure:

```

<ListMetadataFormats>
  <ListFormat>
    <Format>
      <metadataPrefix> oai_dc </metadataPrefix>
      <metadataNamespace> http://www.openarchives.org/OAI/2.0/oai_dc/ </metadataNamespace>
      <schema> http://www.openarchives.org/OAI/2.0/oai_dc.xsd </schema>
    </Format>
  </ListFormat>
  <ListFormat>
    <Format>
      <metadataPrefix> oai_scolomfr </metadataPrefix>
      <metadataNamespace> http://www.lom-fr.fr/scolomfr/utils/xsd/scolomfrv31/ </metadataNamespace>
      <schema> http://www.lom-fr.fr/scolomfr/utils/xsd/scolomfrv31/scolomfr.xsd </schema>
    </Format>
  </ListFormat>
</ListMetadataFormats>

```

Below the XML content, there is a section titled "About the XSLT" which provides information about the XSLT conversion process.

Figure 4 – Entrepôt OAI-PMH d’Édubase

3.4.1.2 Protocole OAI-PMH

OAI-PMH (Open Archives Initiative - Protocol for Metadata Harvesting) signifie le protocole pour la collecte de métadonnées de l’initiative pour les archives ouvertes. Ce protocole est un moyen d’échanger sur Internet des métadonnées entre plusieurs institutions, afin de multiplier les accès aux documents numériques.

3.4.1.3 Mon travail

Mon travail consiste à récupérer tous les fichiers dans l’entrepôt OAI-PMH d’Édubase via un programme Python. Afin d’obtenir les fichiers dans cet entrepôt, nous devons passer par le protocole OAI-PMH.

Sickle^[40] est une bibliothèque client OAI-PMH légère écrite en Python. Elle a été conçue pour récupérer facilement les données des interfaces OAI à la manière de Python.

Il convient de noter ici que Sickle récupère les métadonnées en réponse à des groupes de 100 fichiers. Lorsque la quantité de fichiers cibles est inférieure à 100, un fichier vide sera utilisé à la place. Heureusement, tous les ID de fichiers vides sont vides. Afin de ne pas être affecté par ces fichiers vides, après avoir téléchargé ces fichiers, je lirai les ID de ces fichiers via `xml.etree.ElementTree`^[41] de Python, et nommerai automatiquement les fichiers par leurs ID lors du vidage de ces fichiers, de sorte qu’en conséquence, redondant les fichiers vides sont remplacés (Figure 5).

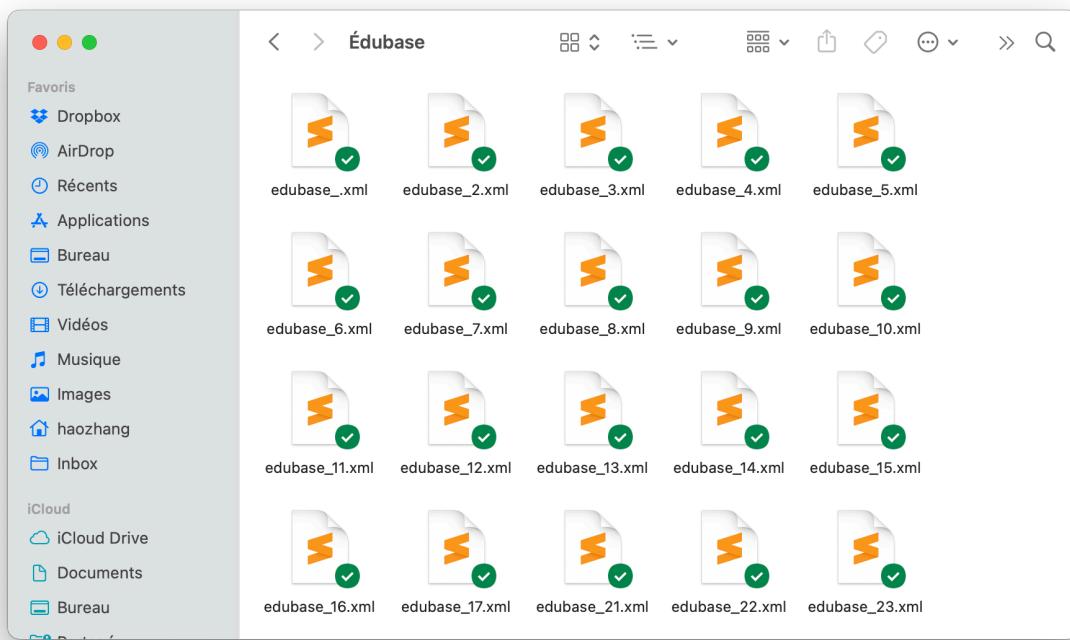


Figure 5 – Fichiers obtenus d’Édubase via OAI-PMH

3.4.2 Construction des ontologies via des fichiers XML

3.4.2.1 Owlready2, un package de programmation orienté ontologie en Python

Owlready2^[42] est un package de programmation orienté ontologie en Python. Il peut charger des ontologies OWL 2.0 en tant qu’objets Python, les modifier, les enregistrer et effectuer des inférences via HermiT (inclus). Owlready permet un accès transparent à l’ontologie OWL.

Owlready2 peut :

- importez des ontologies au format RDF/XML, OWL/XML ou NTriples ;
- manipulez des classes d’ontologie, des instances et des annotations comme des objets Python ;
- ajoutez une méthode Python à la classe d’ontologie ;
- reclassez automatiquement les instances à l’aide du raisonneur HermiT ;
- importez des termes médicaux depuis l’UMLS.

3.4.2.2 Mon travail

Au début de mon stage, les experts de l’équipe du Projet HUMANE ont compilé des corpus dans le domaine de l’HNE, qui se présentent pour la plupart sous forme de cartes mentales. Mon travail consiste à convertir ces cartes mentales en fichiers au format XML via la fonction fournie avec Freeplane^[43] (à gauche de la figure 6), puis à extraire le contenu de chaque nœud de ces fichiers XML via XML.etree.ElementTree ou BeautifulSoup^[44], et suivant la structure de ces nœuds, composant l’ontologie couche par couche. La raison pour laquelle deux manières

différentes sont utilisées pour extraire les nœuds est que le contenu de certains nœuds n'est pas une simple chaîne, mais un contenu plus riche sous forme de HTML (à droite de la figure 6). Alors que l'extraction de contenu de la première manière ne fait attention qu'à la structure du nœud, tandis que l'extraction de contenu de la deuxième manière accorde plus d'attention au contenu du nœud.

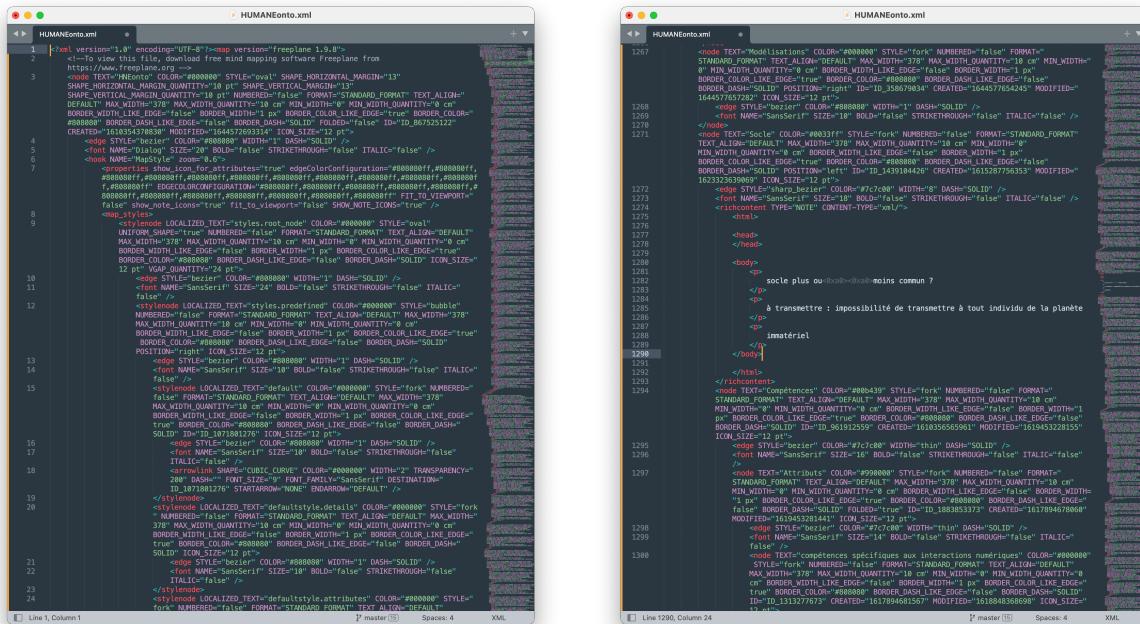


Figure 6 – Carte mentale convertie en structure XML

En raison de la différence entre la structure de la carte mentale et l'ontologie, lorsque je la convertis en ontologie, s'il n'y a pas d'exigence particulière, je convertis d'abord les nœuds de la carte mentale en classes, puis je convertis les classes en attributs ou personnes avec l'avis d'experts. Étant donné que les noms des classes dans l'ontologie ne peuvent pas être répétés, les experts peuvent ne pas tenir compte lors de la construction de la carte du cerveau. Lors de la conversion dans l'ontologie, les nœuds de relecture détruiront la structure normale de la carte du cerveau, et même causeront des problèmes tels que l'héritage cyclique. Cela nécessite de prendre le temps de discuter avec l'auteur original de la carte du cerveau.

Notez également que certains caractères ne sont pas requis par le format URI^[45]. En réponse, j'ai écrit la fonction suivante, en réécrivant les symboles non conformes (Listing 1) tout en préservant autant que possible l'intention originale de l'auteur.

```

1  def no_repeated_spaces(string):
2      pattern = r'\s+'
3      replace = ''
4      return re.sub(pattern, replace, string)
5
6  def no_space(string):
7      return string.strip().replace(" ", " ")
8
9  def no_comparison(string):
10     return string.replace("<", ".:").replace(">", ".:").replace("=", ".:")

```

```

11
12     def no_brackets(string):
13         pattern = r'[\(\) \[\]\{\}\] '
14         replace = '*'
15         return re.sub(pattern, replace, string)
16
17     def no_specials_symbols(string):
18         pattern = r'[\|, \\, \^, \', “\, \”, \@, \#, \%, \=, \,] '
19         replace = '_'
20         return re.sub(pattern, replace, string)
21
22     def IRI_encoder(string):
23         string = no_repeated_spaces(string)
24         string = no_space(string)
25         string = no_comparison(string)
26         string = no_brackets(string)
27         string = no_specials_symbols(string)
28         return string
29

```

Listing 1 – Fonction de conversion de symboles conformes à l'URI

3.4.3 Construction de web sémantique avec la base de données Neo4j et le framework Flask

3.4.3.1 Générer une base de données Neo4j avec une structure de type ontologie

Ma tâche à cette étape est de convertir le fichier XML obtenu via Édubase en l'instruction Cypher correspondante selon sa structure de fichier, et de générer directement la base de données Neo4j. La structure de la base de données Neo4j générée à partir d'un seul fichier XML est illustrée à gauche de la figure 7. La structure de la base de données Neo4j générée par tous les fichiers XML est représentée sur le côté droit de la Figure 7 (limitée à la limite du nombre de nœuds affichés, seul un maximum de 300 nœuds peuvent être affichés).

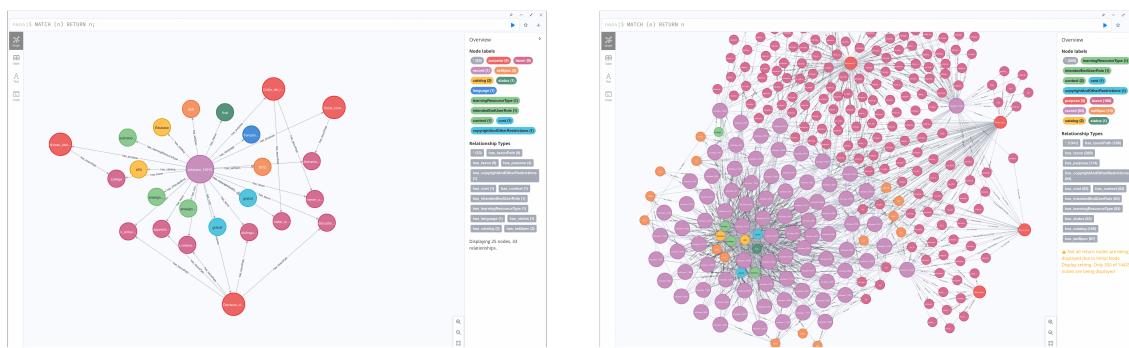


Figure 7 – Structure du fichier XML dans la base de données Neo4j

3.4.3.2 Fonctions de base complètes des pages web via Flask

En raison de contraintes de temps, je n'ai complété que quelques fonctions de pages web. Les pages web peuvent obtenir les dernières informations sur les fichiers dans l'ordre chronologique

(Figure 8, en haut). L'utilisateur peut également cliquer sur le libellé sous le fichier pour obtenir en même temps les informations du fichier portant ce libellé (Figure 8, en bas).

. Esprit critique, es-tu là ?

Comment exercer son esprit critique face aux idées reçues scientifiques ? Ce scénario d'escape game permet de découvrir trois critères essentiels d'évaluation de la fiabilité d'une information ou d'un document concernant une problématique scientifique.

« Le réchauffement climatique est naturel. Vrai ou faux ? » De vos réponses dépendent l'avenir de la planète ! Enquêtez et résolvez trois énigmes, qui vous donneront les clés pour répondre de façon éclairée.»

(EDUBASE_18910)

. Variations du $\delta^{18}\text{O}$ à l'échelle globale

Formation à l'esprit critique en science avec l'exploitation d'enregistrements des teneurs isotopiques de l'oxygène en de multiples points sur Terre pour mettre en

. Escape Game : s'approprier le CDI par le jeu

Séance de découverte du CDI, de son fonctionnement et de ses ressources pour tous les élèves de seconde générale et professionnelle sous forme d'un Escape Game.

- **NIVEAU ÉDUCATIF DÉTAILLÉ**
 - [2de générale et technologique](#)
- **SOCLE COMMUN**
 - [domaine 2.3 : médias, démarches de recherche et de traitement de l'information](#)
- **DOMAINE D'ENSEIGNEMENT**
 - [enseignements du lycée](#)

(EDUBASE_16308)

. Prise de notes sur document numérique (niveau lycée)

Les élèves doivent sélectionner et synthétiser des informations à partir de pages web. Cette séquence pédagogique fait partie des travaux du groupe sur la prise de notes. Outils et méthodologie pour la prise de notes sur document numérique <http://cdi.ac-dijon.fr/spip.php?article284> Documents pour...

- **NIVEAU ÉDUCATIF DÉTAILLÉ**

Figure 8 – Interface pour le web sémantique Flask

Remarque : Le template de web que j'ai utilisé provient de HTML5 UP^[46].

4 Conclusion

Ce stage a été une chance pour moi, après plusieurs mois de recherche afin que je puisse commencer mon deuxième stage formel dans le domaine d'informatique en France. Heureusement, j'ai pu être admis par cette offre d'ELLIADD, et passer une inoubliable durée de 5 mois avec l'équipe d'experts, même à distance.

Le contenu de mon stage est le développement de l'ontologie et ses services intelligents. Le service intelligent fait ici référence à la plate-forme web sémantique. Tout au long du processus, j'ai acquis une compréhension de base et une expérience pratique du concept et du fonctionnement de l'ontologie et du Knowledge Graph, ce qui a élargi mes perspectives de carrière. Non seulement j'ai profité de cette occasion pour apprendre à extraire du contenu pertinent via XML et d'autres types de fichiers, mais j'ai aussi appris à exploiter et à créer des bases de données Neo4j et à créer des pages web via Flask. Cela m'a beaucoup aidé à développer mes compétences professionnelles pour mon avenir. Outre, j'ai réalisé de nombreux défis et percées et reçu une grande satisfaction.

En plus d'améliorer et de renforcer mon niveau technique de programmation en Python, cela a également élargi l'étendue des problèmes que je peux résoudre en utilisant la programmation Python. Mais plus important encore, je peux non seulement acquérir des connaissances de base liées au graphe de connaissances, au web sémantique et à l'ontologie pendant cette période, mais j'ai également acquis des opportunités pratiques significatives et approfondi la plupart de mes connaissances et de ma compréhension du domaine. Cela m'a également donné un fort intérêt pour le travail dans ce domaine.

Après cinq mois de stage, j'ai accumulé une certaine expérience dans l'utilisation de Flask pour développer des pages web et dans l'utilisation des données Neo4j pour stocker et gérer des données, et j'ai une certaine confiance pour faire face aux défis de tels projets à l'avenir. Bien que je n'ai pas eu l'opportunité d'apprendre à développer des bases de données RDF durant ce stage, je suis tout à fait disposé à avoir l'opportunité dans le futur de développer de telles bases de données pour approfondir l'expérience de la différence entre la base de données RDF et les bases de données Neo4j.

J'espère que mon stage pourra jouer un certain rôle de référence pour les futures recherches de mon tuteur de stage, et j'espère contribuer à ma maigre puissance pour la suite de ce projet.

Glossaire français

DNE : la Direction du Numérique pour l'Éducation

DRNE : la Délégation Régionale du Numérique pour l'Éducation de la région académique Bourgogne-Franche-Comté

GIS2if : le Groupement d'Intérêt Scientifique Innovation, Interdisciplinarité, Formation

GTnum : les groupes thématiques numériques

HN : les Humanités Numériques

HNE : les Humanités Numériques et Éducation

Laboratoire ELLIADD : le laboratoire Édition, Littératures, Langages, Informatique, Arts, Didactiques, Discours

Projet HUMANE : le projet HUMAnités Numériques pour l'Éducation

SIC : les Sciences de l'Information et de la Communication

Glossaire anglais

BSD : Berkeley Source Distribution

KG : Knowledge Graph

LPG : Linked Property Graph

n10s : neosemantics

OAI-PMH : Open Archives Initiative Protocol for Metadata Harvesting

OWL : Web Ontology Language

RDF : Resource Description Framework

SKOS : Simple Knowledge Organization System

SMW : Semantic MediaWiki

URI : Uniform Resource Identifier

Références

- [1] https://fr.wikipedia.org/wiki/Knowledge_graph
- [2] <https://www.openarchives.org/pmh>
- [3] <https://eduscol.education.fr/2174/enseigner-et-apprendre-avec-la-recherche-les-groupes-thematiques-numeriques-gtnum>
- [4] <https://gis-2if.shs.parisdescartes.fr>
- [5] <https://discord.com>
- [6] <https://trello.com>
- [7] <https://github.com/RaphaelZH>
- [8] <https://virtualenv.pypa.io/en/latest>
- [9] <http://tetherless-world.github.io/whyis>
- [10] Giachelle F, Dosso D, Silvello G. 2021. Search, access, and explore life science nanopublications on the web. PeerJ Comput. Sci. 7:e335 DOI 10.7717/peerj-cs.335
- [11] <https://blazegraph.com>
- [12] <https://depot.readthedocs.io/en/latest>
- [13] <https://docs.celeryq.dev/en/stable>
- [14] Jim McCusker and Sabbir M. Rashid and Nkechinyere N. Agu and Kristin P. Bennett and Deborah L. McGuinness. 2018. The Whyis Knowledge Graph Framework in Action. SEMWEB. McCusker2018TheWK
- [15] https://www.semantic-mediawiki.org/wiki/Help:Introduction_to_Semantic_MediaWiki
- [16] <https://www.mediawiki.org/wiki/MediaWiki>
- [17] <https://flask.palletsprojects.com/en/2.1.x>
- [18] <https://www.djangoproject.com>
- [19] <https://en.wikipedia.org/wiki/Microframework>
- [20] <https://www.freebsd.org/copyright/freebsd-license>
- [21] <https://jinja.palletsprojects.com/en/3.1.x/intro>
- [22] <https://palletsprojects.com/p/werkzeug>
- [23] <https://wsgi.readthedocs.io/en/latest/what.html>
- [24] <https://flask.palletsprojects.com/en/2.1.x/extensions>
- [25] <https://en.wikipedia.org/wiki/Model-view-controller>
- [26] <https://peps.python.org/pep-0206/#batteries-included-philosophy>
- [27] <https://neo4j.com>
- [28] <https://www.g2.com/categories/rdf-databases>

- [29] <https://neo4j.com/labs/neosemantics>
- [30] <https://dzone.com/articles/sparql-and-cypher>
- [31] <https://neo4j.com/graphgists/graph-databases-rdf-and-linked-data>
- [32] <https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference>
- [33] <https://jbarrasa.com/2018/02/01/neo4j-is-your-rdf-store-part-3-thomson-reuters-openpermid>
- [34] Ezequiel José Veloso Ferreira Moreira and José Carlos Ramalho. 2020. SPARQLing Neo4J. 9th Symposium on Languages, Applications and Technologies (SLATE 2020). 10.4230/OASIcs.SLATE.2020.17. Article No. 17; pp. 17:1-17:10
- [35] <https://neo4j.com/labs/neosemantics/4.0/import>
- [36] <https://neo4j.com/labs/neosemantics/4.0/importing-ontologies>
- [37] <https://www.derivo.de/en/products/graphscale>
- [38] <https://neo4j.com/blog/neo4j-rdf-graph-database-reasoning-engine>
- [39] <http://edubase.eduscol.education.fr>
- [40] <https://sickle.readthedocs.io/en/latest>
- [41] <https://docs.python.org/3/library/xml.etree.elementtree.html>
- [42] <https://owlready2.readthedocs.io/en/v0.37/index.html>
- [43] <https://www.freeplane.org/wiki/index.php/Home>
- [44] <https://beautiful-soup-4.readthedocs.io/en/latest>
- [45] <https://www.w3.org/2011/rdf-wg/wiki/IRIs/RDFConceptsProposal>
- [46] <https://html5up.net>